This manuscript was accepted for publication in Engineering Applications of Artificial Intelligence
https://doi.org/10.1016/j.engappai.2023.106029

Real-Time Measurement-Driven Reinforcement Learning Control Approach for Uncertain Nonlinear Systems ⋆

Mohamed Abouheaf[a], Derek Boase[b], Wail Gueaieb[b,*], Davide Spinello[c], Salah Al-Sharhan[d]

[a] *College of Technology, Architecture and Applied Engineering, Bowling Green State University, Bowling Green, OH, 43403-0001, USA*
[b] *School of Electrical Engineering and Computer Science, University of Ottawa, 800 King Edward Avenue, Ottawa, ON, K1N 6N5, Canada*
[c] *Department of Mechanical Engineering, 161 Louis Pasteur, Ottawa, ON, K1N 6N5, Canada*
[d] *Machine Intelligence Research Labs, , Auburn, WA, 98071-2259, USA*

Abstract

The paper introduces an interactive machine learning mechanism to process the measurements of an uncertain, nonlinear dynamic process and hence advise an actuation strategy in real-time. For concept demonstration, a trajectory-following optimization problem of a Kinova robotic arm is solved using an integral reinforcement learning approach with guaranteed stability for slowly varying dynamics. The solution is implemented using a model-free value iteration process to solve the integral temporal difference equations of the problem. The performance of the proposed technique is benchmarked against that of another model-free high-order approach and is validated for dynamic payload and disturbances. Unlike its benchmark, the proposed adaptive strategy is capable of handling extreme process variations. This is experimentally demonstrated by introducing static and time-varying payloads close to the rated maximum payload capacity of the manipulator arm. The comparison algorithm exhibited up to a seven-fold percent overshoot compared to the proposed integral reinforcement learning solution. The robustness of the algorithm is further validated by disturbing the real-time adapted strategy gains with a white noise of a standard deviation as high as 5%.

Keywords: Optimal control, Adaptive control, Reinforcement learning, Adaptive critics, Model-reference adaptive systems

1. Introduction

Measurement-driven solutions based on adaptive learning concepts are challenged by many factors, such as the need to incorporate the dynamics of the process explicitly into the underlying strategies [1, 2]. Many adaptive approaches have been designed offline and lack the ability to capture high-order model-following dynamics [3–6]. Hence, many adaptive learning approaches employ either complex or computationally expensive algorithms [5–8]. This gets more challenging when adaptive mechanisms are adopted for coupled regulation and optimization missions, where the dimensions of the state and action spaces grow significantly [9–11].

Machine Learning approaches have been employed in many instrumentation applications, such as quality monitoring of laser cladding [12], vortex flow-meter design [13], measurement of residual Oxygen concentration [14], vision-based measurement systems [15], state-of-charge prediction [16], localization of faults and network status detection [17], Parkinson's disease diagnostics [18], machine health monitoring [19], real-time aging prediction of integrated circuits [20], vision systems calibration in welding robots [21], and sleep apnea analysis [22]. Other machine learning forms have adopted Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) networks to solve various optimization problems [23]. A maneuvering system is developed for Unmanned Aerial Vehicles (UAVs) using dynamic inversion and LSTM network approaches [24]. Another adaptive cruise mechanism adopted a transfer learning idea that is based on LSTM networks [25]. The Hierarchical Temporal Memory (HTM) and LSTM network approaches have been adopted to predict short-term arterial traffic flow [26]. A deep neural network that employs a feedback control concept is adopted to solve an intelligent structural control problem [27]. It makes use of a state-selector function to avoid forgetting key states by the neural-networks and hence improve the overall performance. An LSTM-based deep learning approach has been used to predict the vapor mass quantity in the adsorption bed [28]. Other machine learning mechanisms based on LSTM, Bidirectional Long Short-Term Memory (BiLSTM), and Gated Recurrent Unit (GRU) have been employed to improve the efficiency of adsorption cooling systems [29].

Reinforcement learning (RL) is a class of Machine Learning that offers a structured approach to learn the best strategy-to-follow [30–32]. It leverages feedback from an agent's interactions with its environment to either reward or penalize the agent's actions through the utility of a value function [33]. The goal of the agent is to maximize a cumulative sum of the rewards [34]. This class of adaptive systems uses two-step techniques known as policy iteration (PI) and value iteration (VI) [34–37]. For nonlinear applications, Integral Reinforcement Learn-

⋆This work was partially supported by NSERC Grant EGP 537568-2018.
*Corresponding author
Email addresses: mabouhe@bgsu.edu (Mohamed Abouheaf), dboas065@uottawa.ca (Derek Boase), wgueaieb@uottawa.ca (Wail Gueaieb), dspinell@uottawa.ca (Davide Spinello), salah27@ieee.org (Salah Al-Sharhan)

ing (IRL) approaches are adopted to solve optimal control problems [38]. The means of adaptive critics are employed to implement the RL solutions using two neural network structures, namely the actor and critic networks [34]. The adaptive critics approximate the strategy-to-follow using an actor neural network; while the value of applying a certain strategy is approximated by a critic neural network. These approaches have been used to solve cooperative control problems for multi-agent systems communicating over graphs [39–41]. An adaptive Fuzzy-RL mechanism is adopted to control flocking motion of a swarm of robots in [42]. Regression models such as iterative and batch least squares are employed to implement the PI solutions [37, 43]. The adaptive approaches are adopted to control underactuated vehicles and distributed generation sources [44, 45].

Linear Quadratic Tracking Regulators (LQT) provide offline control strategies that solve the optimal tracking control problems. This requires a knowledge of the system dynamics, where the optimal control gains are then applied to the forward evolution of the state [46]. This problem is ubiquitous in modern control applications, namely in intelligent control systems [1]. In order to develop robust adaptive control solutions, it is often desirable to develop a model for the plant, or at least an approximate dynamic model. Although this approach has certain benefits, modeling the dynamics of a system can require assumptions that may narrow the applicability of the model and introduce uncertainty about the dynamic system parameters. Applying model approximations techniques, such as linearization for instance, can lead to a loss of generality. In cases where a dynamic model is available, certain model-reference adaptive control approaches may be considered. These involve backstepping, sliding mode control, and Lyapunov methods, for example [47–54]. Given the dependency of such methods on the dynamics the process, the control strategies inherit such limitations. This can be seen in [49], for example, where the lateral motion of a 5-DoF trailer system is stabilized with a model reference adaptive system (MRAS) using Lyapunov theory [49].

The above mentioned challenges are tackled in this work using an integral adaptive learning approach. Herein, another form of MRAS is proposed for the online control of unknown nonlinear systems. It is then validated using a 6-DoF Kinova robotic arm. The control gains are adapted to reflect the variations in the dynamics of the robotic application. The adaptive learning algorithm actuates the joints following interactive reference-trajectories. It employs a data-driven scheme to determine the variations in the control strategies needed to move the end-effector between the desired positions. The controller relies on an online IRL mechanism with guaranteed stability for slowly varying dynamics. The work contributes an online measurement-driven adaptive learning mechanism that (i) adopts incremental learning capabilities to improve the control strategy in real-time, (ii) avoids incorporating the process and the reference-model dynamics explicitly into the underlying control strategy, (iii) provides a flexible feedback mechanism in terms of the order of the model-following dynamics, and (iv) allows for online approximate solutions for a class of optimal tracking problems. It builds on the contributions of [55] to develop an online IRL control mechanism for nonlinear model-following systems with uncertain dynamics. The work presented herein supports the theoretical findings of [55] with solid data-processing and practical evidence. A data-driven approach is developed for the real-time control of a 6-DoF Kinova robotic arm, as a highly nonlinear dynamic system. It adopts incremental learning features to adapt the strategies without estimating the dynamics of the robotic arm or explicitly expressing the strategies in terms of the reference-trajectory dynamics. In addition, the control structure is modified to accommodate incremental learning capabilities associated with the control strategy (i.e., provide data-driven features to the underlying strategies allowing for online adaptations). The value of using the temporal difference equation in rejecting high-frequency noise is also explored and analyzed in details. Finally, the proposed solution is benchmarked against another model-free approach for dynamic payload and disturbances.

The rest of the paper is organized as follows: The objectives of the adaptive learning problem are highlighted in Section 2. Section 3 introduces a description of the trajectory-tracking optimization problem. Section 4 lays out the mathematical foundation of the temporal difference solution, including a stability analysis of that solution. The IRL solution and its implementation using an adaptive critics technique are introduced in Section 5. The practical experimentation setup and results are discussed in Sections 6 and 7, respectively. Finally, main concluding remarks are highlighted in Section 8.

2. Instrumentation Setup of the Kinova Robotic Arm

The data-driven approach presented herein is applicable to a large-class of nonlinear systems. It is applied to solve the model-following problem of a Kinova JACO manipulator, where each joint is controlled independently of the other. Note that in addition to the nonlinear nature of the dynamics of each joint, it is also time-dependent due to the simultaneous motion of the joints, which adds to the complexity of the control problem. The inherited features in the IRL solution enable the direct use of joint-measurements without estimating the dynamics of the robotic arm. Furthermore, the temporal difference structure of the online IRL solution (i.e., integral temporal difference or Bellman equation) enables robust filtering characteristics for the high-frequency content of the processed signals. This will be highlighted later when presenting the results in Section 7.

The Kinova JACO manipulator is a 6-DoF robot equipped with a two-finger gripper, as shown in Fig. 1. The actuators are interfaced through commercial RS-485 communication protocol providing a data rate of 12 Mbps and a high-level control frequency of 100 Hz. Their low-level control loops operate at 500 MHz. Each servometer is equipped with a position sensor with a resolution of $3,686,400$/turn. Herein, the robot is controlled through the Robot Operating System (ROS).

Each control signal $u_i(t) \in \mathbb{R}$ actuates the motion of joint i (i.e., the position angle $\theta_i(t)$), and the angular velocity is calculated by $\dot{\theta}_i(t) = (\theta_i(t) - \theta_i(t-\nu))/\nu$, where t is a time-index, ν is a sampling-time, and $i \in \{1, 2, 3, 4\}$ represents the joint that is being controlled. Hereafter, index i is omitted for simplicity, since the algorithm is the same for each joint.

Figure 1: Kinova JACO manipulator

3. Problem Formulation

This section introduces the mathematical foundation of the trajectory-tracking optimization problem. The goal is to manipulate the joints of the robotic arm simultaneously to follow a desired trajectory in real-time. The learning scheme decides on the actuation signals of the joints online through adapted strategies to regulate the trajectory tracking-errors between the desired and measured angular positions $\theta^d(t)$ and $\theta(t)$, respectively. Hence, the dynamic model of each joint is given by

$$\dot{\theta} = f\left(\theta(t), u(t)\right). \tag{1}$$

It is assumed here that the drift dynamics of the joint are embedded in unknown function f. The adapted strategy does not require to incorporate such dynamics explicitly in its structure. Herein, the trajectory-tracking problem is formulated as an optimization problem, where the main objective is to select an actuation signal in real-time (i.e., a control strategy that is decided based on a machine learning process) to regulate the trajectory-tracking errors $\varepsilon(t) = \theta^d(t) - \theta(t)$ (i.e., ideally, the aim is to achieve $\lim_{t\to\infty} \|\varepsilon(t)\| = 0$). The control signal due to an attempted strategy π is given by $u^\pi(t) = u^\pi(t-\nu)+\eta^\pi(t)$, where the correction in the actuation signal is represented by $\eta^\pi(t)$ such that $\eta^\pi(t) = \omega_0^\pi\,\varepsilon(t) + \omega_\nu^\pi\,\varepsilon(t-\nu) + \omega_{2\nu}^\pi\,\varepsilon(t-2\nu)$. The notation ν refers to the desired sampling interval needed to collect the real-time measurements. Further, the tuple $\{\omega_0^\pi, \omega_\nu^\pi, \omega_{2\nu}^\pi\}$ defines the control gains of a policy π, which will be determined using the online RL solution. The structure of the correction signal $\eta^\pi(t)$ could vary to reflect high-order error dynamics. This is done by increasing the number of employed error samples (i.e., $\varepsilon(t-o\,\nu), o = 0, 1, \ldots, \mathcal{L}$). Herein, a second-order tracking error system is considered (i.e., $o = 2$), which proved to be sufficient to achieve the optimization goals, as shall be demonstrated later. The signal $\eta^\pi(t)$ decides on the relative angular deviations to apply without any information on the dynamics of the robotic arm. The choice of $o = 2$ means that, this approach can represent as double as the error dynamics of a model-following problem.

The trajectory-tracking problems are mostly solved using adaptive learning approaches, since it is difficult to implement the optimal tracking solutions for complex high-order error dynamical systems [1]. Typically, optimal tracking control problems are solved offline based on the knowledge of the system dynamics. Further, such solutions often start by solving a subset of coupled differential equations offline before solving the remaining subset online [46]. The dynamics of the reference-model must be explicitly embedded in the structure of such solutions, which makes their complexities dependent on the number of controlled joints. Moreover, the solutions do not ensure robustness against varying dynamics of the robotic arm. All of this urges for an innovative model-free adaptive solution that can be realized in real-time. Herein, the proposed IRL scheme is divided into two parts. First, an optimal tracking control setup is considered to develop a temporal difference structure that can be solved online. Then, an adaptive learning scheme based on IRL is considered to solve the trajectory-tracking optimization problem. The solution is realized by sampling and processing the tracking errors of each joint, as illustrated in Fig. 2.

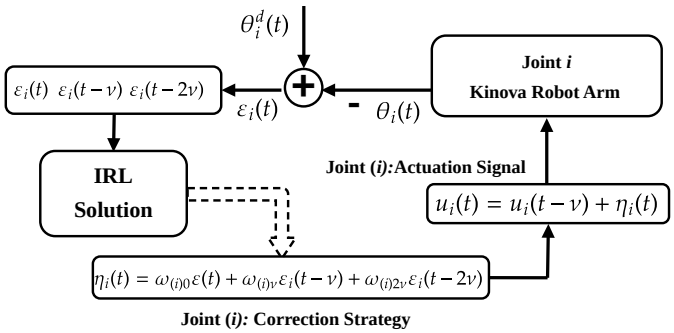


Figure 2: Trajectory-tracking system

4. Optimal Trajectory Tracking

This section introduces the mathematical foundation of the online adaptive learning solution. It relies on an optimal control setup rather than employing an explicit optimal tracking control one. This is done to relax the challenges associated with solving the optimal tracking control problems [46]. However, this strategy needs a proper process to ensure that it does not require information on the robot dynamics. The solution is realized in a measurement-driven manner, where a moving finite storage of tracking-error samples is considered. As highlighted earlier, the number of error samples defines the order of the error dynamics. This structure exhibits an advantage compared to low-order error dynamical systems employed in classical adaptive systems [46, 56]. The goal is to select an optimal strategy $\pi*$ which yields an optimal correction signal $\eta^{\pi*}$ for each joint independently, to follow a desired trajectory. The error samples are stored in a vector $\boldsymbol{X}(t) = [\,\varepsilon(t)\ \ \varepsilon(t-\nu)\ \ \varepsilon(t-2\nu)\,]^T \in \mathbb{R}^3$, which signifies the state vector in an optimal control setup. A convex performance index P, that is inspired by the linear quadratic regulator structure, is employed such that $P^\pi(t) = \int_t^\infty \mathcal{U}(\boldsymbol{X}(\xi), \eta^\pi(\xi))\,d\xi$, where the cost function $\mathcal{U}(\ldots)$ incorporates $\boldsymbol{X}$ and η^π such that $\mathcal{U}(\boldsymbol{X}(\xi), \eta^\pi(\xi)) =$

$\frac{1}{2}\left(\boldsymbol{X}^T(\xi)\,\boldsymbol{Q}\,\boldsymbol{X}(\xi)+\eta^{\pi T}(\xi)\,R\,\eta^{\pi}(\xi)\right)$. The positive definite matrices $\mathbf{0}<\boldsymbol{Q}\in\mathbb{R}^{3\times 3}$ and $0<R\in\mathbb{R}$ are weighting structures. In this particular case, R is a real scalar.

The performance index P evaluates a strategy $\eta^{\pi}(t)$ over a finite interval. The following developments explain: (i) the setup of the implicit optimal control solution; (ii) the conditions needed to develop a candidate solution; (iii) the temporal difference structure employed by the adaptive learning solution; and (iv) the model-free structure of the optimal control strategy. These developments combine findings from the adaptive and optimal control theories.

Theorem 1. *Let a kernel solution function $S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$ be non-negative and $S\left(\mathbf{0}\right)=0$. Thus,*

1. *$S^{*}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$ represents an optimal solution for the Hamilton-Jacobi-Bellman equation (HJB) equation $H(\boldsymbol{X}(t),\boldsymbol{\nabla}S^{*}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right),\eta^{\pi*}(t))=0$.*
2. *$S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$ represents a Lyapunov function.*

Proof. 1. The Hamiltonian for the trajectory-tracking optimization problem decides the optimal policy along the trajectory of the error dynamics $\dot{\boldsymbol{V}}^{\pi}(t)=0$ such that

$$H(\boldsymbol{X}(t),\boldsymbol{\mu}(t),\eta^{\pi}(t))=\boldsymbol{\mu}^{T}(t)\,\dot{\boldsymbol{V}}^{\pi}(t)+\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right),$$

where $\boldsymbol{V}^{\pi}(t)=[\,\boldsymbol{X}^{T}(t)\;\;\eta^{\pi}(t)^{T}\,]^{T}$ and $\boldsymbol{\mu}$ denotes a Lagrange multiplier associated with the constraints $\dot{\boldsymbol{V}}^{\pi}(t)$.

The kernel solution structure $S(\dots)$ is selected to be convex in vector $\boldsymbol{V}^{\pi}(t)$ such that

$$S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=\frac{1}{2}\boldsymbol{V}^{\pi T}(t)\,\boldsymbol{\mathcal{H}}\,\boldsymbol{V}^{\pi}(t), \tag{2}$$

where $\mathbf{0}<\boldsymbol{\mathcal{H}}\equiv\begin{bmatrix}\boldsymbol{\mathcal{H}}_{\boldsymbol{XX}} & \boldsymbol{\mathcal{H}}_{\boldsymbol{X}\eta}\\ \boldsymbol{\mathcal{H}}_{\eta\boldsymbol{X}} & \boldsymbol{\mathcal{H}}_{\eta\eta}\end{bmatrix}\in\mathbb{R}^{4\times 4}$, $\boldsymbol{H}_{\eta\boldsymbol{X}}\in\mathbb{R}^{1\times 3}$ and $\boldsymbol{H}_{\eta\eta}\in\mathbb{R}$.

The relation between the kernel solution and $\boldsymbol{\mu}$ is explained by the Hamilton-Jacobi (HJ) theory [46]. Herein, the Lagrange multiplier is found to be $\boldsymbol{\mu}\equiv\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=\partial S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)/\partial\boldsymbol{V}^{\pi}(t)$. Substituting $\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$ into the Hamiltonian $\mathcal{H}(\dots)$, yields a Bellman equation given by

$$\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)^{T}\;\dot{\boldsymbol{V}}^{\pi}(t)+\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=0. \tag{3}$$

It can be noted that this expression is an infinitesimal representation of $P^{\pi}(t)\overset{\text{def}}{=}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=\int_{t}^{\infty}\boldsymbol{\mathcal{U}}(\boldsymbol{X}(\xi),\eta^{\pi}(\xi))\,d\xi$. Therefore, (3) can be restructured such that

$$H(\boldsymbol{X}(t),\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right),\eta^{\pi}(t))=\dot{S}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)+\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right). \tag{4}$$

Solving $H(\boldsymbol{X}(t),\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right),\eta^{\pi}(t))=0$, while applying the optimal strategy yields the HJB equation. The optimal signal $\eta^{\pi*}$ is derived by applying Bellman's optimality conditions such that $\eta^{\pi*}(t)=\arg\min_{\eta^{\pi}(t)}H(\boldsymbol{X}(t),\boldsymbol{\nabla}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right),\eta^{\pi}(t))$. The optimal solution S^{*} is found by solving the HJB equation given by

$$H(\boldsymbol{X}(t),\boldsymbol{\nabla}S^{*}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right),\eta^{\pi*}(t))=\dot{S}^{*}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right)+\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right)=0. \tag{5}$$

2. Since the kernel solution function S is quadratic and convex, then it represents a Lyapunov candidate function. Using (4) and taking the time-derivative of the kernel solution S yield $\dot{S}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=-\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)\leq 0$. Therefore, $S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$ fulfills the conditions of a Lyapunov function. ■

This mathematical setup solves the optimal tracking control problem through finding the optimal kernel solution S^{*}. However, in order to realize the solution in real-time, a temporal difference structure and an explicit model-free form of the optimal strategy are required. Hence, Theorem 2 builds on the results of Theorem 1 to develop a temporal difference structure that can be employed by the reinforcement learning solution.

Theorem 2. *The kernel solution $S^{*}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right)$ satisfies an integral Bellman optimality equation that is given by*

$$S^{*}\left(\boldsymbol{X}(t),\eta^{\pi*}(t)\right)=\int_{t}^{t+\nu}\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(\xi),\eta^{\pi*}(\xi)\right)\,d\xi+S^{*}\left(\boldsymbol{X}(t+\nu),\eta^{\pi*}(t+\nu)\right), \tag{6}$$

where the optimal policy $\pi*\overset{\text{def}}{=}\{\omega_{0}^{*},\omega_{\nu}^{*},\omega_{2\nu}^{*}\}$ satisfies the stationarity condition of the optimization problem.

Proof. Applying Euler approach on the Bellman equation (3), yields a temporal difference structure given by

$$\frac{S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)-S\left(\boldsymbol{X}(t+\nu),\eta^{\pi}(t+\nu)\right)}{\nu}=\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right).$$

Equivalently, this could be written as

$$S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)=\int_{t}^{t+\nu}\boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(\xi),\eta^{\pi}(\xi)\right)\,d\xi+S\left(\boldsymbol{X}(t+\nu),\eta^{\pi}(t+\nu)\right). \tag{7}$$

This represents a temporal difference form, where Bellman's optimality conditions can be applied to find the optimal strategy. Hence, $\eta^{\pi*}(t)=\arg\min_{\eta^{\pi}(t)}S\left(\boldsymbol{X}(t),\eta^{\pi}(t)\right)$. This and (2) yield

$$\eta^{\pi*}(t)=-\mathcal{H}_{\eta\eta}^{-1}\mathcal{H}_{\eta\boldsymbol{X}}\,\boldsymbol{X}(t). \tag{8}$$

This optimal policy structure $\pi*=-\mathcal{H}_{\eta\eta}^{-1}\mathcal{H}_{\eta\boldsymbol{X}}$, is model-free and relies only on the kernel solution matrix $\mathcal{H}$. Therefore, this matrix is adapted in real-time using the tracking error measurements. Employing this optimal strategy into (7), yields the Integral Bellman optimality equation (6). The optimal correction signal is given by $\eta^{\pi*}(t)=\boldsymbol{\omega}^{*}\boldsymbol{X}(t)$ where the optimal control gains follow $\boldsymbol{\omega}^{*}=[\,\omega_{0}^{*}\;\;\omega_{\nu}^{*}\;\;\omega_{2\nu}^{*}\,]=-\mathcal{H}_{\eta\eta}^{-1}\mathcal{H}_{\eta\boldsymbol{X}}$ ■

The next result discusses the stability of the trajectory-tracking error system following the solution of the Integral Bellman optimality equation (6) using the optimal strategy (8).

Lemma 1. *Let the value of the initial kernel solution S be bounded such that $S\left(\boldsymbol{X}(0), \eta^{\pi}(0)\right) \leq \mathcal{T}$. Given a bounded independent command trajectory $\theta^{d}(t)$, the trajectory tracking error dynamical system is asymptotically stable with $\lim_{t\to\infty} \varepsilon(t) = 0$ and $\lim_{t\to\infty} \dot{S}\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) = 0$.*

Proof. According to Theorem 1, the solution $S\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right)$ is shown to be a Lyapunov function. Then, the inequality given by $S\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) \leq S\left(\boldsymbol{X}(0), \eta^{\pi}(0)\right) \leq \mathcal{T}, \forall t$ holds. This yields, $S\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) \in L_{\infty}$. So, $\varepsilon(t), \varepsilon(t-\nu), \varepsilon(t-2\nu) \in L_{\infty}$ and $\boldsymbol{\mathcal{H}}$ (implicitly signifies a vector of control gains $\boldsymbol{\omega}^{*}$) $\in L_{\infty}$. Since S is proved to be a Lyapunov function and the integral Bellman equation can be formulated as $\int_0^t \dot{S}\left(\boldsymbol{X}(\vartheta), \eta^{\pi}(\vartheta)\right) d\vartheta = S\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) - S\left(\boldsymbol{X}(0), \eta^{\pi}(0)\right)$. Then, $-\int_0^t \dot{S}\left(\boldsymbol{X}(\vartheta), \eta^{\pi}(\vartheta)\right) d\vartheta \leq S\left(\boldsymbol{X}(0), \eta^{\pi}(0)\right)$. This shows that $\dot{S}\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) \in L_{\infty}$, and consequently implies that $\dot{\varepsilon}(t)$, $\dot{\varepsilon}(t-\nu)$, $\dot{\varepsilon}(t-2\nu) \in L_{\infty}$. Using (5) and (8), a conclusion can be made such that $\int_0^t \boldsymbol{X}^{T}(\vartheta)\left(\boldsymbol{Q} + \boldsymbol{\omega}^{*T} R \boldsymbol{\omega}^{*}\right) \boldsymbol{X}(\vartheta)\, d\vartheta \leq S\left(\boldsymbol{X}(0), \eta^{\pi}(0)\right)$, which reveals that $\varepsilon(t)$, $\varepsilon(t-\nu)$, $\varepsilon(t-2\nu) \in L_2$ and $\dot{S}\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) \in L_2$. Applying Barbalat's Lemma [1], yields $\lim_{t\to\infty} \dot{S}\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) = 0$. Thus, the tracking error dynamic system is asymptotically stable and $\lim_{t\to\infty} \varepsilon(t) = 0$. ■

Remark 1. *Adaptive control solutions with time-delays adopt direct and indirect parameter estimation schemes in real-time control strategies. In the model-reference adaptive solution proposed here, the algorithm is based on multi-step time sampling of the tracking error, where the number of steps is left as a user-defined parameter. The time shifting induced by the multi-step sampling naturally incorporates a time-delay into the underlying strategy without requiring additional parameter estimation approaches.*

The above results provide a temporal difference solution framework that uses a model-free strategy to solve an optimal trajectory tracking problem in real-time. It relies on kernel structure (2) and employs a model-free strategy (8) to solve the integral Bellman optimality equation (6). This optimal solution cannot be realized analytically. Hence, a reinforcement learning solution scheme is considered next to realize the solution.

5. Reinforcement Learning Adaptive Solution

The Bellman optimality equation (6) and the optimal model-free control strategy (8) will be used to develop an adaptive learning solution adopting the heuristic form of IRL to control the joints of the Kinova obotic arm, simultaneously in real-time. A value iteration mechanism will be considered to realize the online IRL solution. Then, an adaptive critics structure is employed to approximate the IRL solution in real-time using a gradient-descent technique.

5.1. Online Value Iteration Solution

A simplified value iteration procedure for each joint of the robotic arm is described as follows:

1. Initialize the kernel solution matrix $\mathcal{H}^{0}$, control signal $u(0)$, correction signal $\eta(0)$, and error vector $\boldsymbol{X}(0)$.
2. Start an iterative process $r = 0, 1, 2, \ldots, N$, where r refers to a sequence of adapted or updated policies.
   (a) Solve for the new kernel solution $\mathcal{H}^{r+1}$ such that

$$S^{r+1}\left(\boldsymbol{X}(t), \eta^{r}(t)\right) = \int_t^{t+\nu} \boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(\xi), \eta^{r}(\xi)\right) d\xi + S^{r}\left(\boldsymbol{X}(t+\nu), \eta^{r}(t+\nu)\right). \quad (9)$$

   (b) Improve the correction strategy as

$$\eta^{r+1}(t) = -\left[\mathcal{H}_{\eta\eta}^{-1}\mathcal{H}_{\eta X}\right]^{r+1} \boldsymbol{X}(t). \quad (10)$$

3. Upon convergence of $\|\mathcal{H}^{r}\|$ terminate the adaptation.

This online IRL solution is based on a value iteration mechanism that solves the temporal difference equation (9) and updates the control strategy (10) in the above mentioned designated order. Theorem 3 below verifies the convergence of the adapted kernel solution following this iterative procedure.

Theorem 3. *Let the value iteration solution update the kernel matrix $\mathcal{H}^{r}$ and the optimal strategy $\boldsymbol{\omega}^{r}$ using* (9) *and* (10)*, respectively. Then,*

1. *The value iteration process yields a sequence $0 \leq S^{0} \leq S^{1} \leq S^{2} \leq \cdots \leq S^{*}$ that converges to the optimal solution of* (6)*.*
2. *The strategies advised by* (10) *are stabilizing ones.*

Proof. 1. The initial kernel solution is bounded such that $0 < S^{0}\left(\boldsymbol{X}(\boldsymbol{0}), \eta^{\pi}(\boldsymbol{0})\right) \leq \mathcal{T}$. Thus, according to (9), the equality $S^{r+1}\left(\boldsymbol{X}(t), \eta^{\pi}(t)\right) = \sum_{i=0}^{r} S^{1}\left(\boldsymbol{X}(t+i\nu), \eta^{\pi}(t+i\nu)\right) - \sum_{i=1}^{r} S^{0}\left(\boldsymbol{X}(t+i\nu), \eta^{\pi}(t+i\nu)\right)$ holds. This leads to a non-decreasing sequence (i.e., $0 \leq S^{0} \leq S^{1} \leq \cdots \leq S^{r} \leq S^{r+1}, \forall r$). The trajectory-tracking error dynamical system is shown to be asymptotically stable. Therefore, the cumulative cost is bounded (i.e., $0 < \int_0^{\infty} \boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(\xi), \eta^{\pi}(\xi)\right) d\xi \leq \bar{\mathcal{T}}$). This leads to a converging sequence such that $0 \leq S^{0} \leq S^{1} \leq \cdots \leq S^{r} \leq S^{r+1} \leq \mathcal{T} + \bar{\mathcal{T}}, \forall r$. Accordingly, the value iteration procedure results in a sequence $0 \leq S^{0} \leq S^{1} \leq \cdots \leq S^{*}$, where S^{*} is the solution of Bellman optimlaity equation (6).

2. The integral Bellman optimality equation $S^{r}\left(\boldsymbol{X}(t), \eta^{r}(t)\right) - S^{r}\left(\boldsymbol{X}(t+\nu), \eta^{r}(t+\nu)\right) = \int_t^{t+\nu} \boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(\xi), \eta^{r}(\xi)\right) d\xi$ employs the optimal strategies $\eta^{r}, \forall r, \nu$ given by (10). This and the stability results yield

$$\lim_{\xi\to\infty} S^{r}\left(\boldsymbol{X}(\xi), \eta^{r}(\xi)\right) = 0 \leq \cdots \leq S^{r}\left(\boldsymbol{X}(t+2\nu), \eta^{r}(t+2\nu)\right) \leq S^{r}\left(\boldsymbol{X}(t+\nu), \eta^{r}(t+\nu)\right) \leq S^{r}\left(\boldsymbol{X}(t), \eta^{r}(t)\right).$$

   Therefore, the strategies $\mu^{r}, \forall r$ are stabilizing and the resulting sequence of updated strategies can be written as

$$S^{0}\left(\boldsymbol{X}(t), \eta^{0}(t)\right) \leq S^{1}\left(\boldsymbol{X}(t+\nu), \eta^{1}(t+\nu)\right) \leq S^{2}\left(\boldsymbol{X}(t+2\nu), \eta^{2}(t+2\nu)\right) \leq \cdots \leq S^{r}\left(\boldsymbol{X}(t+r\nu), \eta^{r}(t+r\nu)\right).$$

■

Remark 2. *Generally, Lyapunov solutions for model-following problems require the knowledge of the process dynamics and of the desired trajectory dynamics. IRL is adopted in optimization problems written in a temporal difference form, as for example induced by the integral Bellman equation in a discrete-time form. Temporal difference forms with function approximation are the core of families of model-free solution methods for optimization problems. Herein, the simultaneous solution of* (9) *and* (10) *provides a novel algorithmic approach to the optimal tracking problem based on the computational form of the IRL.*

5.2. Adaptive Critics Implementation

Neural networks are adopted to implement the value iteration solution. This is done using critic and actor neural networks to approximate the kernel function and associated optimal strategy. These structures are adapted in real-time using data measured along the trajectory of the robotic arm system. The kernel solution value function is approximated such that

$$\hat{S}\left(\boldsymbol{X}(t), \hat{\eta}(t)\right) = \frac{1}{2} \boldsymbol{V}^T(t)\, \boldsymbol{\Omega}_c\, \boldsymbol{V}(t),$$

where $\boldsymbol{\Omega}_c$ represents the critic weights, $\hat{\eta}(t)$ is the approximation of correction control signal, and $\boldsymbol{V}^{\pi}(t) = [\, \boldsymbol{X}^T(t) \;\; \hat{\eta}(t)\,]^T$. The critic weights $\boldsymbol{\Omega}_c > \boldsymbol{0}$ approximate the kernel solution matrix $\boldsymbol{\mathcal{H}} > \boldsymbol{0}$ which is adapted by the value iteration process. This structure is inspired by that of the kernel matrix $\boldsymbol{\mathcal{H}}$.

The structure of the actor neural network follows the form of the optimal strategy (10) such that

$$\hat{\eta}(t) = \boldsymbol{\Omega}_a\, \boldsymbol{X}(t),$$

where the actor neural network weights $\boldsymbol{\Omega}_a$ approximate the optimal control gains $\omega^* = [\, \omega_0^* \;\; \omega_\nu^* \;\; \omega_{2\nu}^*\,]$.

Herein, a gradient-descent approach is employed to adapt the actor and critic weights. Thus, the adaptation error structures of the critic and actor are inspired by the value iteration procedure given by (9) and (10), respectively. The tuning error associated with adapting the critic weights is given by $E_{Critic} = \frac{1}{2}\left(\hat{S}\left(\boldsymbol{X}(t), \hat{\eta}(t)\right) - \tilde{S}(t)\right)^2$, where $\tilde{S}(t)$ is defined by $\tilde{S}(t) = \boldsymbol{\mathcal{U}}\left(\boldsymbol{X}(t), \hat{\eta}(t)\right) + \hat{S}\left(\boldsymbol{X}(t+\nu), \hat{\eta}(t+\nu)\right)$. Therefore, the critic tuning law follows

$$\boldsymbol{\Omega}_c^{(r+1)} = \boldsymbol{\Omega}_c^{(r)} - \alpha_c\, E_{Critic}^{(r)} \boldsymbol{V}^{\pi}\, \boldsymbol{V}^{\pi T}, \tag{11}$$

where $0 < \alpha_c < 1$ is an adaptation rate for the critic weights. Similarly, the tuning error associated with adapting the actor weights is formulated as $E_{Actor} = \frac{1}{2}\left(\hat{\eta} - \tilde{u}\right)^2$, where $\tilde{u} = -\boldsymbol{\Omega}_{c\hat{\eta}\hat{\eta}}^{-1}\, \boldsymbol{\Omega}_{c\hat{\eta}\boldsymbol{X}}$. Thus, the resulting actor adaption law is given by

$$\boldsymbol{\Omega}_a^{(r+1)} = \boldsymbol{\Omega}_a^{(r)} - \alpha_a\, E_{Actor}^{(r)}\, \boldsymbol{X}^T, \tag{12}$$

where $0 < \alpha_a < 1$ is the learning rate of the actor weights. The detailed steps of the online adaptive critics implementation of the IRL solution are listed in Algorithm 1.

Remark 3. *Herein, the problem is formulated as deterministic, consistently with the literature on reinforcement learning solutions of optimal control problem with model-free setups, where optimal control policies are learned by relying on measurements along a specific system's trajectory, eliminating the necessity of the system's drift dynamics/model which may be unknown or highly uncertain [46]. Additive noise is used as disturbance in various scenarios below to test and illustrate the robustness of the adaptive learning solutions.*

Algorithm 1 Online Adaptive Critics Solution

Input:
Total number of adaption steps N
Sampling interval ν
Adaptation rates α_a and α_c
Weighting matrices $\boldsymbol{Q}$ and R
Convergence threshold σ and a time-window of length L to check convergence

Output:
Tuned actor and critic weights (i.e., $\boldsymbol{\Omega}_a^{(t+\ell\nu)}$ and $\boldsymbol{\Omega}_c^{(t+\ell\nu)}$, $\ell = 0, 1, \ldots, N$)

Define the desired trajectory $\theta^d(t), \forall t$
Initialize the angular position $\theta(0)$, control signal $u(0)$, actor weights $\boldsymbol{\Omega}_a^{(0)}$, and critic weights $\boldsymbol{\Omega}_c^{(0)}$
Calculate the error vector $\boldsymbol{X}(0)$
$\ell \leftarrow 0$
Convergence_Condition $\leftarrow$ False
while $\ell < N$ **and** Convergence_Condition = False **do**
    Compute the correction control signal $\hat{\eta}(\ell\nu)$
    Adjust the control signal $u(\ell\nu) + \hat{\eta}(\ell\nu)$ and then actuate each joint
    Observe $\theta((\ell+1)\,\nu)$
    Use $\theta^d((\ell+1)\,\nu)$ to find $\varepsilon(t + (\ell+1)\,\nu)$
    Calculate $\boldsymbol{\mathcal{U}}(\boldsymbol{X}(\ell\,\nu), \hat{\eta}(\ell\nu))$ and $S\left(\boldsymbol{X}(\ell\,\nu), \hat{\eta}(\ell\nu)\right)$
    Find $\boldsymbol{X}((\ell+1)\,\nu)$ and $\hat{\eta}((\ell+1)\,\nu)$. Hence, get $S\left(\boldsymbol{X}((\ell+1)\,\nu, \hat{\eta}((\ell+1)\nu)\right)$
    Find the target values of the critic and actor neural network approximators $\tilde{S}(\ell\,\nu)$ and $\tilde{\eta}(\ell\nu)$
    Adapt the critic weights $\Theta_c^{(\ell+1)}$ ▷ use tuning law (11)
    Adapt the actor weights $\Theta_a^{(\ell+1)}$ ▷ use tuning law (12)
    if $\ell > L$ **and** $\left\|\boldsymbol{\Omega}_c^{(\ell+1-l)} - \boldsymbol{\Omega}_c^{(\ell-l)}\right\| \leq \sigma$, $\forall l \in \{0, 1, \ldots, L\}$, **then**
        $\boldsymbol{\Omega}_c^{(*)} \leftarrow \boldsymbol{\Omega}_c^{(\ell+1)}$
        Convergence_Condition $\leftarrow$ True
    end if
    $\ell \leftarrow \ell + 1$
end while
return $\boldsymbol{\Omega}_a^{(t+\ell\nu)}$ and $\boldsymbol{\Omega}_c^{(t+\ell\nu)}$, for $\ell = 0, 1, \ldots, N$

6. Experimental Setup

Five experiments are carried out to evaluate the controller's performance under various conditions. In the first four experiments, the control scheme is applied to simultaneously track the nominal trajectories of the four joints (base, shoulder, elbow,

and wrist) under four different conditions. In these experiments, the initial joint positions are taken as $\boldsymbol{\theta}(0) = [\,0 \;\; 90 \;\; 180 \;\; -135 \;\; 180 \;\; 0\,]^T$ (degrees). Note that the last two joints of the robot are not controlled. The first experiment implements the control algorithm in an ideal case where there is no payload and no dynamic disturbances. The response to this experiment is used as a baseline for the comparison with the other responses, along with the nominal trajectories. To test the algorithm under a different payload, the second experiment is conducted while a gripper, mounted at the robot's end-effector, holds a constant payload of 3 lb, which represents approximately 91% of the arm's total payload capacity. The purpose of the third experiment is to assess the controller's performance in the face of sudden payload changes. It is conducted by starting with no payload and then abruptly adding a 2.5 lb sandbag to the gripper around the halfway mark of the experiment. To evaluate the disturbance rejection capacity of the proposed solution, a fourth experiment is carried out where the actor weights are synthetically disturbed by additive noise drawn from the normally distributed random variable $M \sim N(0, \sigma)$ during the beginning of the experiment. This experiment is repeated while varying the duration and standard deviation of the weight disturbances. The values used are tabulated in Table 1. Such a scenario, demonstrates the robustness of the algorithm to variations in initial weights and shows a strong ability to adapt the weights as needed. Furthermore, the experiment mimics measurement noise that may be imposed on or passed to the actuation signals. It is important to note that this synthetic disturbance is superimposed to the actual measurement noise intrinsic to the instrumentation of the manipulator's arm. The fifth experiment tests the controller's ability to compensate for a time-varying payload by controlling a single-joint (the elbow) while the gripper gradually lifts a sandbag off the top of a table so that the payload carried by the robot gradually increases to the full-weight of the sandbag (2.5 lb). The initial joint positions in this time-varying payload experiment are set to $\boldsymbol{\theta}(0) = [\,90 \;\; 143 \;\; -45 \;\; -135 \;\; 180 \;\; 0\,]^T$ (degrees). A brief summary of the five experiments is provided in Table 2. Experiment 4 is the only one that was not conducted on the real robot. It is rather run on ROS/Gazebo simulation platform to protect the robot from any erratic behaviors that may arise due to the injected noise.

Table 1: Disturbance experiment parameters

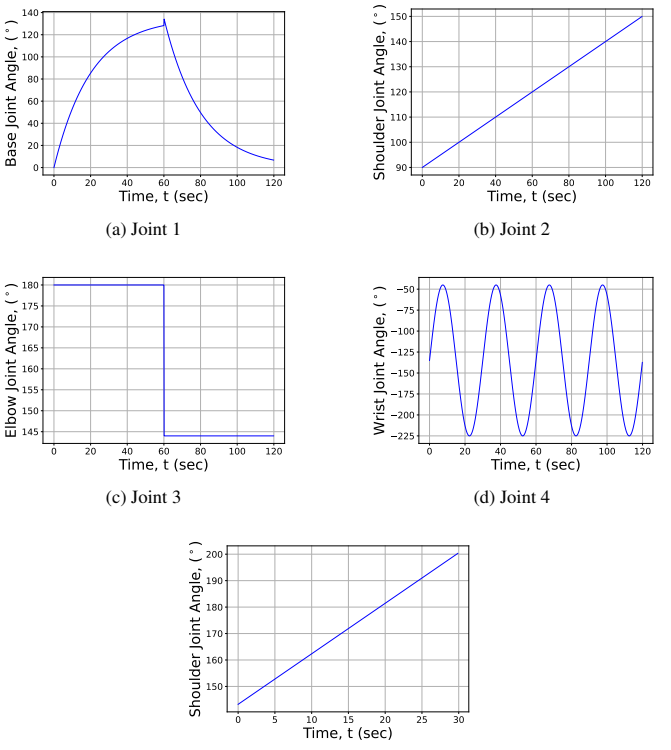
Duration (%)	20	40	60	100
Variance, σ^2	0.025	0.025	0.020	0.0125

The nominal joint trajectories for the first four experiments are shown in Figs. 3(a) to 3(d), while that of the fifth experiment is shown in Fig. 3(e). This collection of reference trajectories provides variety of model-following dynamics to test, including slow dynamics and faster dynamics (the latter represented by the high frequency sinusoidal signal). Further, sharp and nonlinear forms of trajectories are considered. The goal is to test the robustness of the IRL solution under co-exiting interacting dynamic trajectories. The base joint trajectory in

Table 2: Summary of the experiments

Experiment	Description
1	No payload, no disturbance
2	Constant payload of 3 lb throughout the experiment
3	Abrupt payload of 2.5 lb halfway throughout the experiment
4	Random distrurbance of actor weights at the beginning of the experiment
5	Time-varying payload

Fig. 3(a) represents an exponential growth for a predefined duration of three time-constants followed by an exponential decay for another duration of three time-constants. This trajectory is used to observe the system's ability to follow a path with time-varying position and velocity. The linear reference trajectory of the shoulder shown in Fig. 3(b) probes the system's ability to follow a time-varying angular position while maintaining a constant velocity. The elbow's target trajectory in Fig. 3(c) aims to analyze the ability of the system to maintain a constant angular position for a period of time before reacting to a step change. A notable difference between the elbow's trajectory and the other trajectories is the discontinuity of the former. The step change in the desired elbow's angular position should force the IRL algorithm to extrapolate and deliberately choose points that are not specified by the given values. Finally, the reference trajectory of the wrist shown in Fig. 3(d) investigates the algorithm's ability to track sinusoidal signals.



(a) Joint 1

(b) Joint 2

(c) Joint 3

(d) Joint 4

(e) Joint 2

Figure 3: (a)–(d): Nominal trajectories for experiments 1 to 4. (e): Elbow's reference trajectory for experiment 5.

The high-level control loop operates at a rate of 8 Hz for the IRL algorithm with $N = 960$ and $\nu = 0.125$ s, where here N and ν represent the discrete-time index and the cycle period of the actuation system, respectively. In all test cases, the adaptation rates for the actor and critic weights are selected to be $\alpha_a = 0.01$ and $\alpha_c = 0.05$, respectively. The initial critic matrices are randomly generated for the first test case and then held constant throughout the remainder of the test cases for consistency purposes. Since the joint encoders sampling frequency and the actuation frequency are 50 Hz and 8 Hz, respectively, the corresponding time delay is negligible. The high-level ROS control between the external computer and the Kinova manipulator is secured through a USB-B connection while the embedded low-level actuator controls leverage the RS-485 protocol.

The critic weighting matrices for the four joints are set to

$$\boldsymbol{\Omega}_{c_1} = \begin{bmatrix} 0.80350 & 0.30937 & 0.84494 & 0.71454 \\ 0.30937 & 0.21330 & 0.31157 & 0.36979 \\ 0.84494 & 0.31157 & 1.09927 & 0.53435 \\ 0.71454 & 0.36979 & 0.53435 & 0.92855 \end{bmatrix},$$

$$\boldsymbol{\Omega}_{c_2} = \begin{bmatrix} 0.42471 & 0.49183 & 0.54214 & 0.52932 \\ 0.49183 & 0.66047 & 0.31157 & 0.59427 \\ 0.54214 & 0.31157 & 1.15278 & 0.89470 \\ 0.52932 & 0.59427 & 0.89470 & 1.05989 \end{bmatrix},$$

$$\boldsymbol{\Omega}_{c_3} = \begin{bmatrix} 0.55195 & 0.39823 & 0.37661 & 0.25486 \\ 0.39823 & 0.51538 & 0.42652 & 0.33598 \\ 0.37661 & 0.42652 & 0.40267 & 0.35263 \\ 0.25486 & 0.33598 & 0.35263 & 0.48076 \end{bmatrix},$$

$$\boldsymbol{\Omega}_{c_4} = \begin{bmatrix} 1.52075 & 0.78373 & 1.29889 & 1.07203 \\ 0.78373 & 0.93637 & 0.54343 & 0.28335 \\ 1.29889 & 0.54343 & 1.41173 & 0.91172 \\ 1.07203 & 0.28335 & 0.91172 & 1.2790 \end{bmatrix}.$$

The initial actor weights are decided using the above critic matrices such that $\Omega_a^0 = -\mathcal{H}_{\eta\eta}^{-1}\mathcal{H}_{\eta X} + Y$, where $Y \sim N(0, 0.1)$. The random variable is added to the base, shoulder, and elbow joints, to show robustness to the initial conditions of the actor weights and to more readily demonstrate the adaptability of the IRL. The values of $\boldsymbol{Q}$ and R are quoted below for completeness.

$$\boldsymbol{Q}_1 = \begin{bmatrix} 0.51503 & 0.25789 & 0.06581 \\ 0.25789 & 0.19214 & 0.07471 \\ 0.06581 & 0.07471 & 0.03784 \end{bmatrix}, \quad R_1 = 0.07451,$$

$$\boldsymbol{Q}_2 = \begin{bmatrix} 0.85038 & 0.59431 & 0.47996 \\ 0.59431 & 0.51992 & 0.24568 \\ 0.47996 & 0.24568 & 0.38590 \end{bmatrix}, \quad R_2 = 0.00876,$$

$$\boldsymbol{Q}_3 = \begin{bmatrix} 0.74237 & 0.51836 & 0.63923 \\ 0.51836 & 0.41698 & 0.46758 \\ 0.63923 & 0.46758 & 0.60572 \end{bmatrix}, \quad R_3 = 0.49363.$$

$$\boldsymbol{Q}_4 = \begin{bmatrix} 0.56665 & 0.36332 & 0.53481 \\ 0.36332 & 0.47523 & 0.37991 \\ 0.53481 & 0.37991 & 0.5266 \end{bmatrix}, \quad R_4 = 0.019686.$$

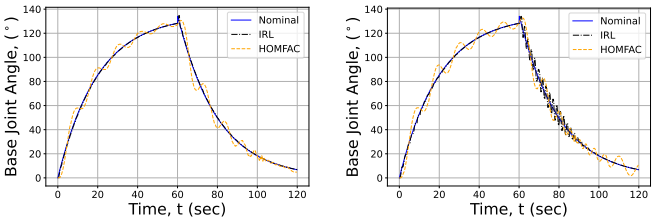
To put the performance of the IRL algorithm in perspective, we compare it with a high-order model-free adaptive control (HOMFAC) scheme presented in [57], which is an improved version of the algorithm proposed in [58]. The adaptive learning solution differs in the structure when compared with that of the HOMFAC approach. The adaptive solution employs adaptable strategies unlike the HOMFAC approach, where fixed control gains are considered. Furthermore, the reinforcement learning strategy captures explicit high-order error dynamics, while the order of error dynamics is controllable. This is unlike the HOMFAC case, where explicit zero-order error dynamics are utilized into the control strategy. This means that the adaptive learning solution has more capacity to react to the variations in the error patterns. The simulation cases in Section 7 highlight the impact of such differences. Just like the IRL algorithm, the HOMFAC technique is applied to each of the four joints. Both algorithms are implemented in Python 2.7 and interfaced with the robot through ROS Melodic. The parameter values used for the HOMFAC algorithm are the same as those presented in [57] and are tabulated in Table 3. Parameters α, η, λ, μ, and ρ, dictate the adaptation properties of the estimator $\phi(t)$ and calculation of the control action $u(t)$. Parameters ϕ_i^0, $i = 1, 2, 3, 4$, represent the initial conditions of the joint estimators. These values are chosen experimentally. In order to successfully implement the HOMFAC algorithm on the Kinova JACO arm, the frequency is reduced to 5 Hz. Attempts to implement the HOMFAC algorithm with higher frequencies induced non-convergent responses.

Table 3: Parameters used for the HOMFAC algorithm

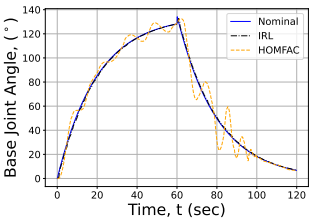
Parameter	Value	Parameter	Value
α	[1/2 1/4 1/8 1/8]	η	0.8
λ	0.1	μ	0.01
ρ	0.8	ϕ_1^0	15
ϕ_2^0	15	ϕ_3^0	25
ϕ_4^0	25		

7. Results and Discussion

The joint trajectories achieved with the IRL and HOMFAC algorithms are logged at run-time and plotted against the nominal trajectories for comparison. The results for the first three experiments are illustrated in Figs. 4 to 7. It is observed that the IRL algorithm is able to converge rapidly to the reference signals, which is not always the case for the HOMFAC technique. These figures clearly demonstrate the superiority of the IRL algorithm over the HOMFAC. This is more evident in the cases where there is a sudden or/and continuous changes in the reference trajectories (e.g., Figs. 6 and 7). As a matter of fact, in some experiments, the HOMFAC exhibits excessive oscillations and even divergence, as in Fig. 6 and Fig. 7(c), respectively, while the IRL algorithm shows a smooth and rapid convergence. Quantitatively, Fig. 6(c) shows a maximum elbow joint overshoot using the HOMFAC algorithm of approximately 28% as opposed to approximately 4% for the IRL. For the purpose of readability and completeness, the adaptations of the actuator gains are presented in the Appendix.

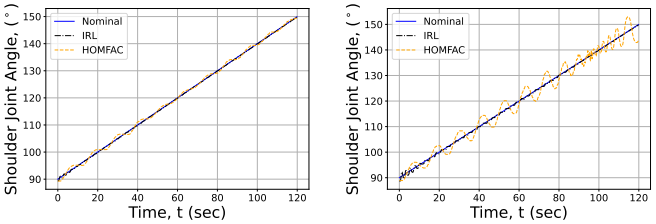


(a) Experiment 1 (b) Experiment 2

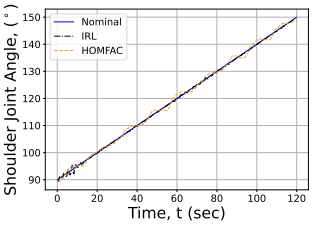


(c) Experiment 3

Figure 4: Base trajectories for the first three experiments.

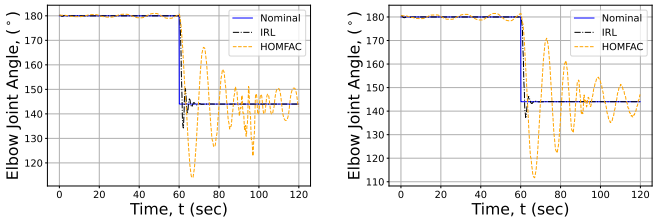


(a) Experiment 1 (b) Experiment 2

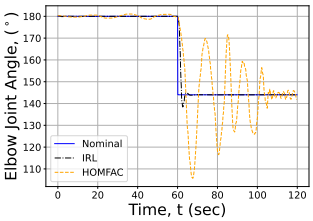


(c) Experiment 3

Figure 5: Shoulder trajectories for the first three experiments.



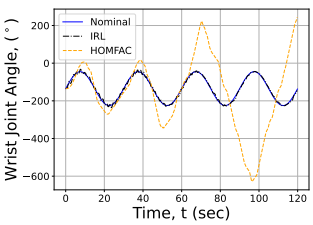
(a) Experiment 1 (b) Experiment 2



(c) Experiment 3

Figure 6: Elbow trajectories for the first three experiments.

(a) Experiment 1 (b) Experiment 2

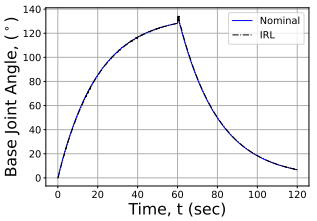


(c) Experiment 3

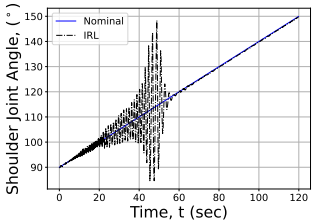
Figure 7: Wrist trajectories for the first three experiments.

The tracking performance of the IRL algorithm for experiment 4 is shown in Fig. 8. The adaptations of the elbow actor gains are depicted in Fig. 9. For completeness, the gain figures for the rest of the joints can be found in Appendix A. Despite the extra noise injected in the first 24 s of the experiment, the IRL algorithm maintained a trajectory profile which oscillates closely around the reference-signal during this period of time before it rapidly converges after that. This is particularly clear for joints 3 and 4 (the oscillations in joint 1 are too small to be noticed). The significant oscillations in the shoulder joint between the 20 s and 60 s point marks are due to the dependence of motion between joints 2 and 3. The axes of rotation for these two joints are parallel to one another, implying that any jerk in joint 2 influences joint 3. This is clearly observed by considering the response of joints 2 and 3 between the 45 s and 50 s point marks. During this interval, an increase in oscillation is observed in both joints before the behavior is rectified by the IRL algorithm and the trajectory error is successfully reduced preceding the step change in joint 3 at the 60 s mark. The actor's effort in counter-acting the noise is manifested in the dynamic behavior of its gains during the first 24 s. They smoothly converge soon after that. For readability, the results of the experiments with longer disturbance periods are included in Appendix B.

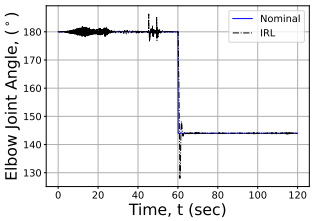
The robot's initial configuration and the results for experiment 5 are visualized in Figs. 10 and 11, respectively. These results are consistent with those observed earlier in that, the HOMFAC algorithm seems to ill-cope with continuously changing dynamics (due to the time-varying payload in this case). Nonetheless, the IRL method easily converges despite a few minor initial oscillation cycles. As expected, the learning process of the IRL algorithm remains active while the payload is vary-



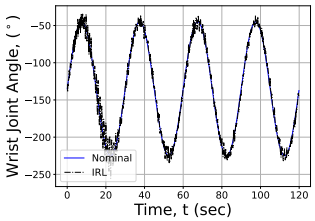
(a) Joint 1



(b) Joint 2

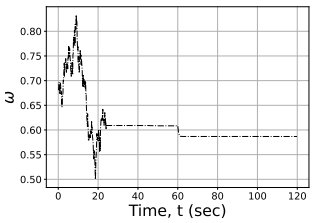


(c) Joint 3

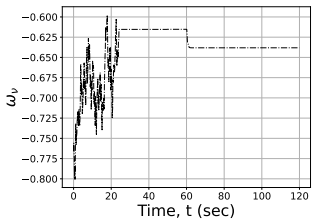


(d) Joint 4

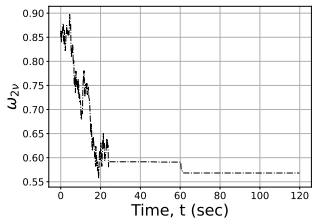
Figure 8: Joint trajectories for experiment 4.



(a) Joint 3–Actor 1, ω



(b) Joint 3–Actor 2, ω_ν



(c) Joint 3–Actor 3, $\omega_{2\nu}$

Figure 9: Adaptation of the elbow actor gains for experiment 4.

ing and stabilizes right after it reaches a constant value. This is shown in the variation of the actor gains plotted in Fig. 12.

Figure 10: Robot initial position for experiment 5.

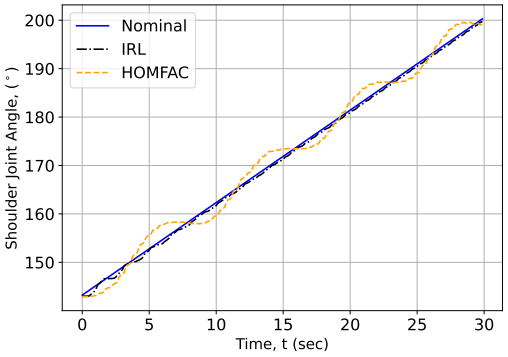
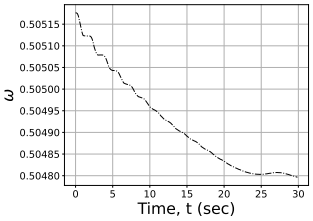
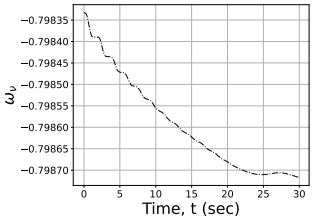


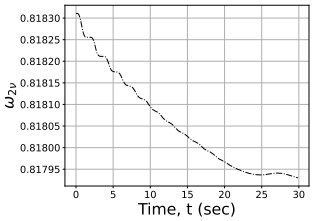
Figure 11: Shoulder trajectories for experiment 5.



(a) Joint 2–Actor 1, ω



(b) Joint 2–Actor 2, ω_ν



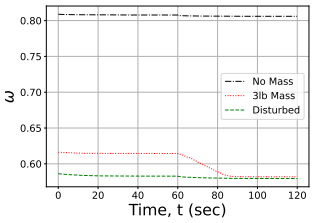
(c) Joint 2–Actor 3, $\omega_{2\nu}$

Figure 12: Adaptation of the shoulder actor gains for experiment 5.

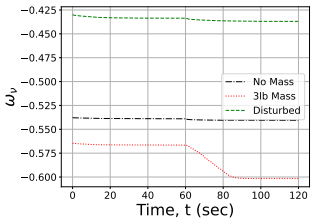
8. Conclusion

The paper discusses the need for reliable real-time processing schemes of sensor readings using robust machine learning algorithms. To that end, an integral reinforcement learning approach is developed for the control of a class of nonlinear systems. This is accomplished in real-time without prior knowledge of the robot dynamics or explicitly incorporating the desired trajectory in the adapted strategies. The solution is realized using a value iteration process. The convergence and stability characteristics of the adaptive learning solution are formally analyzed. The proposed technique is demonstrated on a 6-DoF Kinova robotic arm and is compared with another model-free controller (HOMFAC). Results show the superiority of the former approach under various dynamics and reference signals, including high-frequency measurement noise.

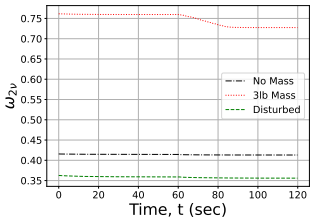
Appendix A. Actors Adaptations for Experiments 1-3



(a) Joint 1–Actor 1, ω

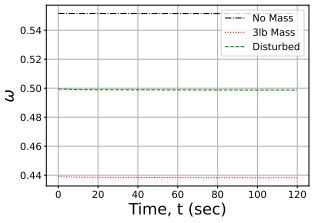


(b) Joint 1–Actor 2, ω_v

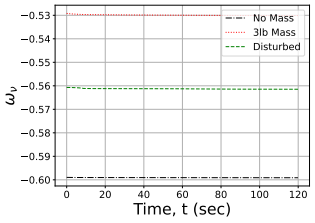


(c) Joint 1–Actor 3, ω_{2v}

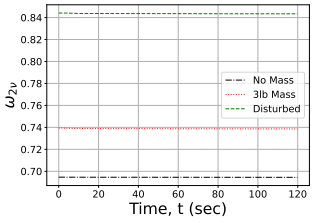
Figure A.13: Adaptation of the base actor gains for the first three experiments.



(a) Joint 2–Actor 1, ω

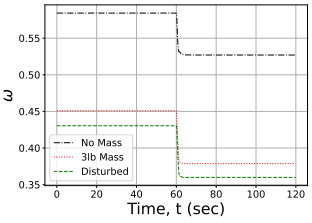


(b) Joint 2–Actor 2, ω_v

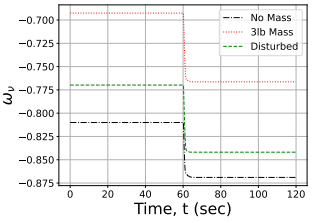


(c) Joint 2–Actor 3, ω_{2v}

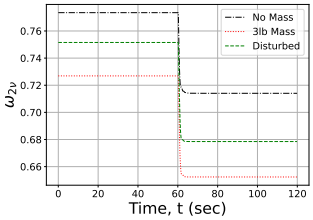
Figure A.14: Adaptation of the shoulder actor gains for the first three experiments.



(a) Joint 3–Actor 1, ω

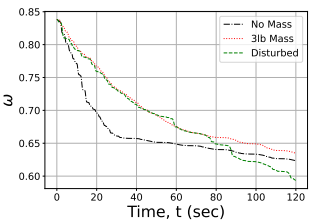


(b) Joint 3–Actor 2, ω_v

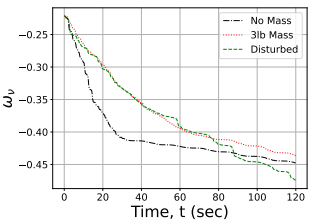


(c) Joint 3–Actor 3, ω_{2v}

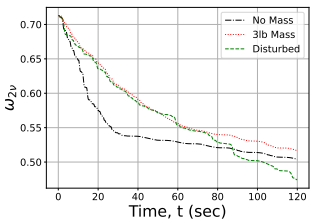
Figure A.15: Adaptation of the elbow actor gains for the first three experiments.



(a) Joint 4–Actor 1, ω



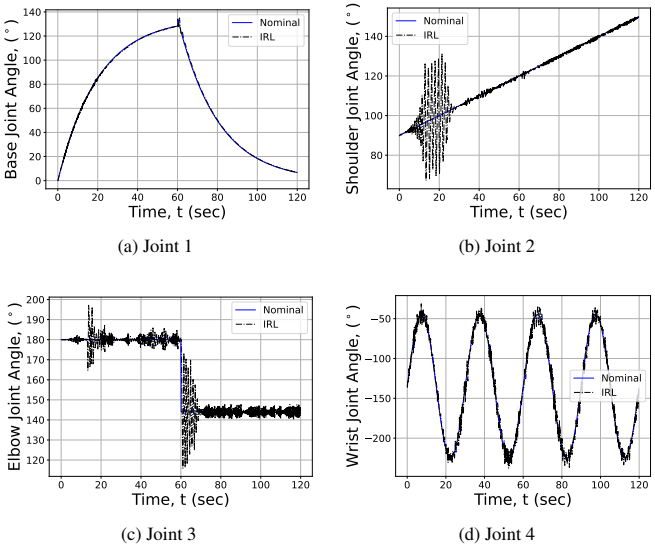
(b) Joint 4–Actor 2, ω_v



(c) Joint 4–Actor 3, ω_{2v}

Figure A.16: Adaptation of the wrist actor gains for the first three experiments.

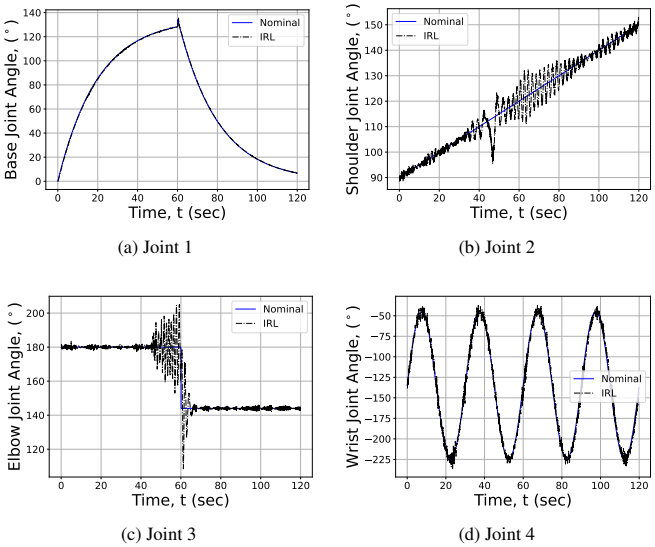
Appendix B. Supplementary Data for Noise Experiments



(a) Joint 1 (b) Joint 2

(c) Joint 3 (d) Joint 4

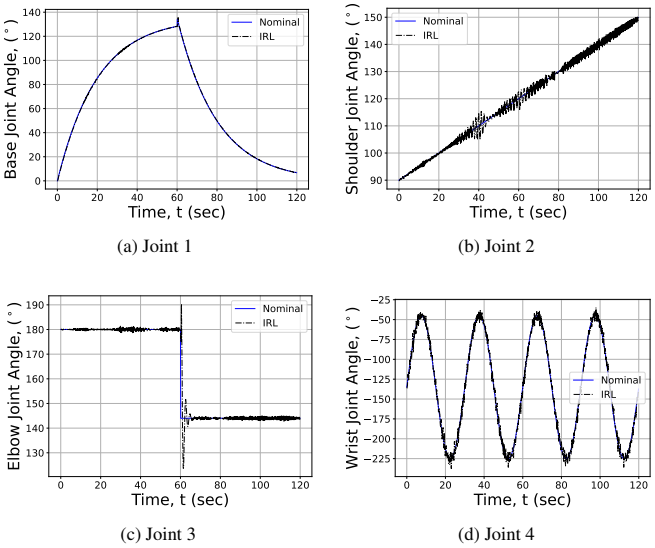
Figure B.17: Joint trajectories for experiment 4 with noise added to actor weights over the first 40% of the experiment.



(a) Joint 1 (b) Joint 2

(c) Joint 3 (d) Joint 4

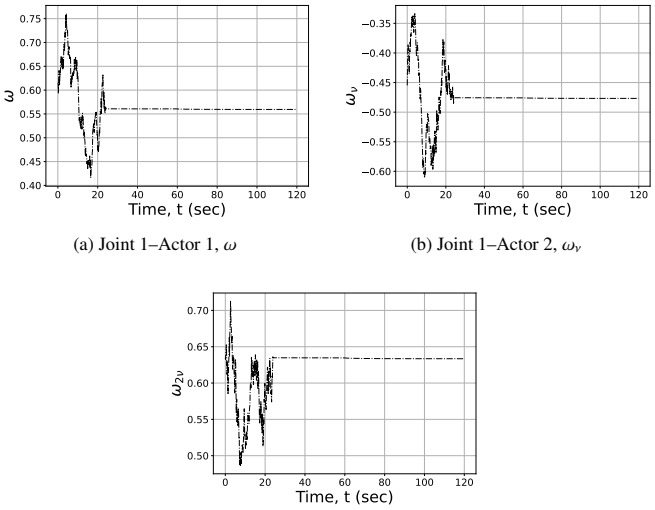
Figure B.18: Joint trajectories for experiment 4 with noise added to actor weights over the first 60% of the experiment.



(a) Joint 1 (b) Joint 2

(c) Joint 3 (d) Joint 4

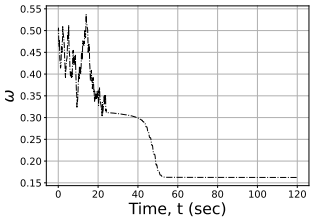
Figure B.19: Joint trajectories for experiment 4 with noise added to actor weights over the experiment.



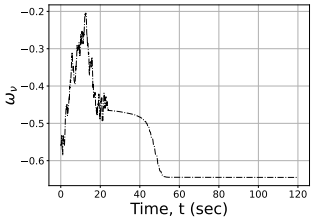
(a) Joint 1–Actor 1, ω (b) Joint 1–Actor 2, ω_ν

(c) Joint 1–Actor 3, $\omega_{2\nu}$

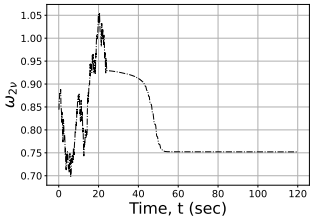
Figure B.20: Adaptation of the base actor gains for experiment 4..



(a) Joint 2–Actor 1, ω

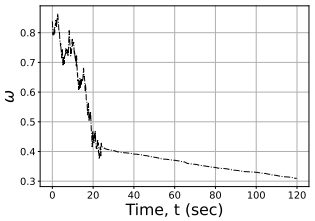


(b) Joint 2–Actor 2, ω_ν

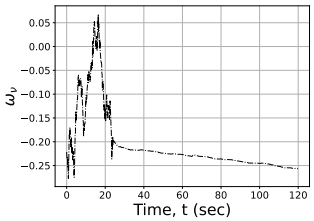


(c) Joint 2–Actor 3, $\omega_{2\nu}$

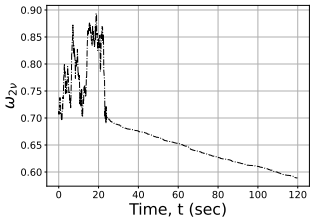
Figure B.21: Adaptation of the shoulder actor gains for experiment 4.



(a) Joint 4–Actor 1, ω



(b) Joint 4–Actor 2, ω_ν



(c) Joint 4–Actor 3, $\omega_{2\nu}$

Figure B.22: Adaptation of the wrist actor gains for experiment 4.

References

[1] K. J. ström, B. Wittenmark, Adaptive Control, Courier Corporation, 2013.

[2] M. V. de Paula, T. A. d. S. Barros, A sliding mode ditc cruise control for srm with steepest descent minimum torque ripple point tracking, IEEE Transactions on Industrial Electronics 69 (1) (2022) 151–159. doi:10.1109/TIE.2021.3050349.

[3] X. Wang, Y. Li, Z. Quan, J. Wu, Optimal trajectory-tracking guidance for reusable launch vehicle based on adaptive dynamic programming, Engineering Applications of Artificial Intelligence 117 (2023) 105497. doi:https://doi.org/10.1016/j.engappai.2022.105497.
URL https://www.sciencedirect.com/science/article/pii/S0952197622004870

[4] H. Peng, F. Li, J. Liu, Z. Ju, A symplectic instantaneous optimal control for robot trajectory tracking with differential-algebraic equation models, IEEE Transactions on Industrial Electronics 67 (5) (2020) 3819–3829. doi:10.1109/TIE.2019.2916390.

[5] C. K. Verginis, C. P. Bechlioulis, A. G. Soldatos, D. Tsipianitis, Robust trajectory tracking control for uncertain 3-dof helicopters with prescribed performance, IEEE/ASME Transactions on Mechatronics (2022) 1–11doi:10.1109/TMECH.2021.3136046.

[6] W. He, X. Tang, T. Wang, Z. Liu, Trajectory tracking control for a three-dimensional flexible wing, IEEE Transactions on Control Systems Technology (2022) 1–8doi:10.1109/TCST.2021.3139087.

[7] P. Li, S. Wang, H. Yang, H. Zhao, Trajectory tracking and obstacle avoidance for wheeled mobile robots based on empc with an adaptive prediction horizon, IEEE Transactions on Cybernetics (2021) 1–10doi:10.1109/TCYB.2021.3125333.

[8] N. Wang, C. K. Ahn, Coordinated trajectory-tracking control of a marine aerial-surface heterogeneous system, IEEE/ASME Transactions on Mechatronics 26 (6) (2021) 3198–3210. doi:10.1109/TMECH.2021.3055450.

[9] R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction, 2nd Edition, Second, MIT Press, Massachusetts, 1998.

[10] J. Cheng, Y. Kang, B. Xin, Q. Zhang, K. Mao, S. Zhou, Time-varying trajectory tracking formation h control for multiagent systems with communication delays and external disturbances, IEEE Transactions on Systems, Man, and Cybernetics: Systems (2021) 1–13doi:10.1109/TSMC.2021.3095850.

[11] Z. Pan, Z. Sun, H. Deng, D. Li, A multilayer graph for multiagent formation and trajectory tracking control based on mpc algorithm, IEEE Transactions on Cybernetics (2021) 1–12doi:10.1109/TCYB.2021.3119330.

[12] I.-H. Kao, Y.-W. Hsu, Y. H. Lai, J.-W. Perng, Laser cladding quality monitoring using coaxial image based on machine learning, IEEE Transactions on Instrumentation and Measurement 69 (6) (2020) 2868–2880. doi:10.1109/TIM.2019.2926878.

[13] D. Thummar, J. Reddy, V. Arumuru, Machine learning for vortex flowmeter design, IEEE Transactions on Instrumentation and Measurement (2021) 1–1doi:10.1109/TIM.2021.3128692.

[14] J. He, C. Song, Q. Luo, L. Lan, C. Yang, W. Gui, Noise-robust self-adaptive support vector machine for residual oxygen concentration measurement, IEEE Transactions on Instrumentation and Measurement 69 (10) (2020) 8474–8485. doi:10.1109/TIM.2020.2987049.

[15] T. Fedullo, D. Cassanelli, G. Gibertoni, F. Tramarin, L. Quaranta, G. de Angelis, L. Rovati, A machine learning approach for a vision-based van-herick measurement system, in: 2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2021, pp. 1–6. doi:10.1109/I2MTC50364.2021.9459946.

[16] Y. Li, M. Maleki, S. Banitaan, M. Chen, Data-driven state of charge estimation of li-ion batteries using supervised machine learning methods, in: 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), 2021, pp. 873–878. doi:10.1109/ICMLA52953.2021.00144.

[17] A. R. Mohammed, S. A. Mohammed, D. Côté, S. Shirmohammadi, Machine learning-based network status detection and fault localization, IEEE Transactions on Instrumentation and Measurement 70 (2021) 1–10. doi:10.1109/TIM.2021.3094223.

[18] A. Talitckii, A. Anikina, E. Kovalenko, A. Shcherbak, O. Mayora, O. Zimniakova, E. Bril, M. Semenov, D. V. Dylov, A. Somov, Defining optimal exercises for efficient detection of parkinson's disease using machine learning and wearable sensors, IEEE Transactions on Instrumentation and Measurement 70 (2021) 1–10. doi:10.1109/TIM.2021.3097857.

[19] C. Zhu, Z. Chen, R. Zhao, J. Wang, R. Yan, Decoupled feature-temporal cnn: Explaining deep learning-based machine health monitoring, IEEE Transactions on Instrumentation and Measurement 70 (2021) 1–13. doi:10.1109/TIM.2021.3084310.

[20] K. Huang, X. Zhang, N. Karimi, Real-time prediction for ic aging based on machine learning, IEEE Transactions on Instrumentation and Measurement 68 (12) (2019) 4756–4764. doi:10.1109/TIM.2019.2899477.

[21] Y. Zou, R. Lan, An end-to-end calibration method for welding robot laser vision systems with deep reinforcement learning, IEEE Transactions on Instrumentation and Measurement 69 (7) (2020) 4270–4280. doi:10.1109/TIM.2019.2942533.

[22] M. Bahrami, M. Forouzanfar, Sleep apnea detection from single-lead ecg: A comprehensive analysis of machine learning and deep learning algorithms, IEEE Transactions on Instrumentation and Measurement (2022) 1–1doi:10.1109/TIM.2022.3151947.

[23] A. Sherstinsky, Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network, Physica D: Nonlinear Phenomena 404 (2020) 132306. doi:https://doi.org/10.1016/j.physd.2019.132306.
URL https://www.sciencedirect.com/science/article/pii/S0167278919305974

[24] C. Su, X. Wang, L. Shen, H. Yu, Adaptive uav maneuvering control system based on dynamic inversion and long-short-term memory network, in: 2020 Chinese Automation Congress (CAC), 2020, pp. 6880–6885. doi:10.1109/CAC51589.2020.9327738.

[25] J. Zhou, J. Wan, F. Zhu, Transfer learning based long short-term memory car-following model for adaptive cruise control, IEEE Transactions on Intelligent Transportation Systems 23 (11) (2022) 21345–21359. doi:10.1109/TITS.2022.3184290.

[26] J. Mackenzie, J. F. Roddick, R. Zito, An evaluation of htm and lstm for short-term arterial traffic flow prediction, IEEE Transactions on Intelligent Transportation Systems 20 (5) (2019) 1847–1857. doi:10.1109/TITS.2018.2843349.

[27] H. Radmard Rahmani, G. Chase, M. Wiering, C. Könke, A framework for brain learning-based control of smart structures, Advanced Engineering Informatics 42 (2019) 100986. doi:https://doi.org/10.1016/j.aei.2019.100986.
URL https://www.sciencedirect.com/science/article/pii/S1474034619305592

[28] D. Skrobek, J. Krzywanski, M. Sosnowski, A. Kulakowska, A. Zylka, K. Grabowska, K. Ciesielska, W. Nowak, Prediction of sorption processes using the deep learning methods (long short-term memory), Energies 13 (24) (2020). doi:10.3390/en13246601.
URL https://www.mdpi.com/1996-1073/13/24/6601

[29] D. Skrobek, J. Krzywanski, M. Sosnowski, A. Kulakowska, A. Zylka, K. Grabowska, K. Ciesielska, W. Nowak, Implementation of deep learning methods in prediction of adsorption processes, Advances in Engineering Software 173 (2022) 103190. doi:https://doi.org/10.1016/j.advengsoft.2022.103190.
URL https://www.sciencedirect.com/science/article/pii/S0965997822000977

[30] R. S. Sutton, A. G. Barto, R. J. Williams, Reinforcement learning is direct adaptive optimal control, IEEE Control Systems Magazine 12 (2) (1992) 19–22.

[31] M. Zolfpour-Arokhlo, A. Selamat, S. Z. Mohd Hashim, H. Afkhami, Modeling of route planning system based on q value-based dynamic programming with multi-agent reinforcement learning algorithms, Engineering Applications of Artificial Intelligence 29 (2014) 163–177. doi:https://doi.org/10.1016/j.engappai.2014.01.001.
URL https://www.sciencedirect.com/science/article/pii/S0952197614000086

[32] V. Samsonov, K. Ben Hicham, T. Meisen, Reinforcement learning in manufacturing control: Baselines, challenges and ways forward, Engineering Applications of Artificial Intelligence 112 (2022) 104868. doi:https://doi.org/10.1016/j.engappai.2022.104868.
URL https://www.sciencedirect.com/science/article/pii/S0952197622001130

[33] F. AlMahamid, K. Grolinger, Autonomous unmanned aerial vehicle navigation using reinforcement learning: A systematic review, Engineering Applications of Artificial Intelligence 115 (2022) 105321. doi:https://doi.org/10.1016/j.engappai.2022.105321.
URL https://www.sciencedirect.com/science/

article/pii/S095219762200358X

[34] D. Bertsekas, J. Tsitsiklis, Neuro-Dynamic Programming, 1st Edition, Athena Scientific, Massachusetts, 1996.

[35] H. Liu, Q. Cheng, J. Xiao, L. Hao, Performance-based data-driven optimal tracking control of shape memory alloy actuated manipulator through reinforcement learning, Engineering Applications of Artificial Intelligence 114 (2022) 105060. doi:https://doi.org/10.1016/j.engappai.2022.105060.
URL https://www.sciencedirect.com/science/article/pii/S0952197622002238

[36] A. Wasala, D. Byrne, P. Miesbauer, J. O'Hanlon, P. Heraty, P. Barry, Trajectory based lateral control: A reinforcement learning case study, Engineering Applications of Artificial Intelligence 94 (2020) 103799. doi:https://doi.org/10.1016/j.engappai.2020.103799.
URL https://www.sciencedirect.com/science/article/pii/S0952197620301858

[37] L. Buşoniu, D. Ernst, B. De Schutter, R. Babuška, Online least-squares policy iteration for reinforcement learning control, in: Proceedings of the 2010 American Control Conference, 2010, pp. 486–491. doi:10.1109/ACC.2010.5530856.

[38] M. Abouheaf, W. Gueaieb, Model-free adaptive control approach using integral reinforcement learning, in: 2019 IEEE International Symposium on Robotic and Sensors Environments (ROSE), 2019, pp. 1–7. doi:10.1109/ROSE.2019.8790432.

[39] M. I. Abouheaf, F. L. Lewis, K. G. Vamvoudakis, S. Haesaert, R. Babuska, Multi-agent discrete-time graphical games and reinforcement learning solutions, Automatica 50 (12) (2014) 3038–3053. doi:https://doi.org/10.1016/j.automatica.2014.10.047.
URL https://www.sciencedirect.com/science/article/pii/S0005109814004282

[40] M. Abouheaf, F. Lewis, M. Mahmoud, D. Mikulski, Discrete-time dynamic graphical games: Model-free reinforcement learning solution, Control Theory and Technology 13 (1) (2015) 55–69.

[41] M. Abouheaf, W. Gueaieb, Multi-agent synchronization using online model-free action dependent dual heuristic dynamic programming approach, in: 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 2195–2201. doi:10.1109/ICRA.2019.8794438.

[42] S. Qu, M. Abouheaf, W. Gueaieb, D. Spinello, An adaptive fuzzy reinforcement learning cooperative approach for the autonomous control of flock systems, in: 2021 International Conference on Robotics and Automation (ICRA), 2021, pp. 8927–8933. doi:10.1109/ICRA48506.2021.9561204.

[43] R. Srivastava, R. Lima, K. Das, A. Maity, Least square policy iteration for ibvs based dynamic target tracking, in: 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 2019, pp. 1089–1098. doi:10.1109/ICUAS.2019.8798242.

[44] M. Abouheaf, N. Q. Mailhot, W. Gueaieb, D. Spinello, Guidance mechanism for flexible-wing aircraft using measurement-interfaced machine-learning platform, IEEE Transactions on Instrumentation and Measurement 69 (7) (2020) 4637–4648. doi:10.1109/TIM.2020.2985553.

[45] M. Abouheaf, W. Gueaieb, A. Sharaf, Load frequency regulation for multi-area power system using integral reinforcement learning, IET Generation, Transmission & Distribution 13 (19) (2019) 4311–4323. arXiv:https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-gtd.2019.0218, doi:https://doi.org/10.1049/iet-gtd.2019.0218.
URL https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-gtd.2019.0218

[46] F. L. Lewis, D. Vrabie, V. L. Syrmos, Optimal Control, John Wiley & Sons, 2012.

[47] R. Byrne, C. Abdallah, Design of a model reference adaptive controller for vehicle road following, Mathematical and Computer Modelling 22 (4) (1995) 343–354. doi:https://doi.org/10.1016/0895-7177(95)00143-P.
URL https://www.sciencedirect.com/science/article/pii/089571779500143P

[48] J. Moore, R. Tedrake, Adaptive control design for underactuated systems using sums-of-squares optimization, in: 2014 American Control Conference, 2014, pp. 721–728. doi:10.1109/ACC.2014.6859508.

[49] S. Vempaty, E. Lee, Y. He, Model-reference based adaptive control for enhancing lateral stability of car-trailer systems, in: ASME International Mechanical Engineering Congress and Exposition, Vol. 50664, American Society of Mechanical Engineers, 2016, p. V012T16A021.

[50] R. Ben Amor, S. Elloumi, Decentralized model reference adaptive control for interconnected robotic systems, in: 2017 18th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), 2017, pp. 235–240. doi:10.1109/STA.2017.8314907.

[51] X. Zuo, J.-w. Liu, X. Wang, H.-q. Liang, Adaptive pid and model reference adaptive control switch controller for nonlinear hydraulic actuator, Mathematical Problems in Engineering 2017 (2017).

[52] Z. Shi, L. Zhao, Robust model reference adaptive control based on linear matrix inequality, Aerospace Science and Technology 66 (2017) 152–159. doi:https://doi.org/10.1016/j.ast.2017.03.017.
URL https://www.sciencedirect.com/science/article/pii/S1270963816309683

[53] J. Hu, G. Feng, Distributed tracking control of leader–follower multi-agent systems under noisy measurement, Automatica 46 (8) (2010) 1382–1387.

[54] H. Chen, Y. Peng, D. Zhang, S. Xie, H. Yan, Dynamic positioning for underactuated surface vessel via l1 adaptive backstepping control, Transactions of the Institute of Measurement and Control 43 (2) (2021) 355–370. arXiv:ttps://doi.org/10.1177/0142331220952960, doi:10.1177/0142331220952960.
URL https://doi.org/10.1177/0142331220952960

[55] M. Abouheaf, W. Gueaieb, D. Spinello, S. Al-Sharhan, A data-driven model-reference adaptive control approach based on reinforcement learning, in: 2021 IEEE International Symposium on Robotic and Sensors Environments (ROSE), 2021, pp. 1–7. doi:10.1109/ROSE52750.2021.9611772.

[56] J. Chen, J. Wang, W. Wang, Model reference adaptive control for a class of aircraft with actuator saturation, in: 2018 37th Chinese Control Conference (CCC), 2018, pp. 2705–2710. doi:10.23919/ChiCC.2018.8484160.

[57] J. Xu, N. Lin, R. Chi, Improved high-order model free adaptive control, in: 2021 IEEE 10th Data Driven Control and Learning Systems Conference (DDCLS), 2021, pp. 704–708. doi:10.1109/DDCLS52934.2021.9455488.

[58] R. Chi, Z. Hou, S. Jin, B. Huang, Computationally efficient data-driven higher order optimal iterative learning control, IEEE Transactions on Neural Networks and Learning Systems 29 (12) (2018) 5971–5980. doi:10.1109/TNNLS.2018.2814628.